===

SHORT QUESTIONS

Define Computer.

A computer is an electronic data processing device. It reads data processing and produces results accurately at a very high speed.

What is algorithm and what is the role of algorithm in problem solving?

Algorithm:

Algorithm means method, procedure, technique or plan. It is a step-by-step problem solving method. It is easy to understand and follow. It is a set of steps that clearly defines a sequence of operations to solve a problem.

Role of Algorithm in Problem Solving:

- Algorithm plays an important role in computer programming.
- Computer programming is the process of taking an algorithm and coding it in a programming language.
- Formulating an algorithm is the first step for developing a computer program.

What is a flowchart?

Flowchart is a diagrammatic representation of algorithm. It describes what operations are required to solve a given problem.

What are the advantages of using flowcharts?

- Flowchart illustrates the sequence of operations to be performed to solve a problem.
- Programmers draw flowcharts before writing computer programs.
- Flowchart provides an easy way to analyze and find solutions of problems.
- Once, the flowchart is drawn, it becomes very easy to write the program in any high level language.
- Flowchart is very helpful in communicating the problem solving method to other people.
- It also helps in finding and removing errors in computer programs.

Draw any four graphical symbols used in flowchart and explain them.

Flowchart symbols have specific meaning and are connected by arrows indicating the flow from one step to another.

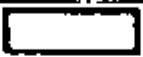
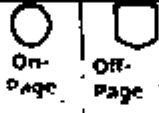
Flow line:

It is a line with arrow head. It is used to connect various flowchart symbols. It indcates the flow of control.

Start/Stop Symbol:

It is a rounded rectangular shaped. It is used to indicate the start or end of a flowchart. We can only write the words START or STOP inside this symbol. A flowchart can only have one start but it may have many ends.

Input/Output Symbol:

It is represented by parallelogram. It indicate input or output operations. It contains the word READ or INPUT along with the variables for input operation or PRINT for OUTPUT along with the output data for output operation.

Processing Symbol:
It is represented by a rectangular block. It is used for data processing operation. All the calculations appear inside the processing symbol, such as "SUM=A+B". Variables are also initialized inside the process symbol such as "K=1".

Decision Symbol:
A decision or branching point. Lines representing different decisions emerge from different points of the diamond.

Table 1-1 Flowchart symbols

Symbol Description	Symbol Shape	Symbol Description	Symbol Shape
Flow Line	→	Process	▭
Start/Stop	⬭	Decision	◇
Input/Output	▱	Connector	On-Page ○ Off-Page ⬠

What is defining the problem?
Defining the problem is first stage of problem solving. It's very important to understand the problem before the programmer starts working on its solution. The following are the steps to properly define and understand the problem.

- Carefully read the problem to understand what it tells.
- Find out what the problem asks to do.
- What information can be obtained from the problem?
- What is required to be calculated as the solution of the problem?

LONG QUESTIONS

Describe the steps involved in problem solving?
The following five steps are involved in problem solving on computer.

1) Defining the Problem:
Defining the problem is first stage of problem solving. It's very important to understand the problem before the programmer starts working on its solution. The following are the steps to properly define and understand the problem.

- Carefully read the problem to understand what it tells.
- Find out what the problem asks to do.
- What information can be obtained from the problem?
- What is required to be calculated as the solution of the problem?

ختم نبوت ﷺ زندہ باد　　　　عظمت صحابہ زندہ باد

السلام علیکم ورحمۃ اللہ وبرکاتہ:

معزز ممبران: آپ کا وٹس ایپ گروپ ایڈ من "**اردو بکس**" آپ سے مخاطب ہے۔

آپ تمام ممبران سے گزارش ہے کہ:

- گروپ میں صرف PDF کتب پوسٹ کی جاتی ہیں لہذا کتب کے متعلق اپنے کمنٹس / ریویوز ضرور دیں۔ گروپ میں بغیر ایڈ من کی اجازت کے کسی بھی قسم کی (اسلامی وغیر اسلامی، اخلاقی، تحریری) پوسٹ کرنا سختی سے منع ہے۔
- گروپ میں معزز، پڑھے لکھے، سلجھے ہوئے ممبرز موجود ہیں اخلاقیات کی پابندی کریں اور گروپ رولز کو فالو کریں بصورت دیگر معزز ممبرز کی بہتری کی خاطر ریموو کر دیا جائے گا۔
- کوئی بھی ممبر کسی بھی ممبر کو انباکس میں میسیج، مس کال، کال نہیں کرے گا۔ رپورٹ پر فوری ریموو کر کے کاروائی عمل میں لائے جائے گی۔
- ہمارے کسی بھی گروپ میں سیاسی و فرقہ واریت کی بحث کی قطعاً کوئی گنجائش نہیں ہے۔
- اگر کسی کو بھی گروپ کے متعلق کسی قسم کی شکایت یا تجویز کی صورت میں ایڈ من سے رابطہ کیجئے۔
- سب سے اہم بات:

گروپ میں کسی بھی قادیانی، مرزائی، احمدی، گستاخِ رسول، گستاخِ امہات المؤمنین، گستاخِ صحابہ و خلفائے راشدین حضرت ابو بکر صدیق، حضرت عمر فاروق، حضرت عثمان غنی، حضرت علی المرتضیٰ، حضرت حسنین کریمین رضوان اللہ تعالیٰ اجمعین، گستاخ اہلبیت یا ایسے غیر مسلم جو اسلام اور پاکستان کے خلاف پراپیگنڈا میں مصروف ہیں یا ان کے روحانی وذہنی سپورٹرز کے لئے کوئی گنجائش نہیں ہے لہذا ایسے اشخاص بالکل بھی گروپ جوائن کرنے کی زحمت نہ کریں۔ معلوم ہونے پر فوراً ریموو کر دیا جائے گا۔

- تمام کتب انٹرنیٹ سے تلاش / ڈاؤنلوڈ کر کے **فری آف کاسٹ** وٹس ایپ گروپ میں شیئر کی جاتی ہیں۔ جو کتاب نہیں ملتی اس کے لئے معذرت کر لی جاتی ہے۔ جس میں محنت بھی صَرف ہوتی ہے لیکن ہمیں آپ سے صرف دعاؤں کی درخواست ہے۔
- عمران سیریز کے شوقین کیلئے علیحدہ سے عمران سیریز گروپ موجود ہے۔
- **لیڈیز کے لئے الگ گروپ کی سہولت موجود ہے جس کے لئے ویریفکیشن ضروری ہے۔**
- اردو کتب / عمران سیریز یا سٹڈی گروپ میں ایڈ ہونے کے لئے ایڈ من سے وٹس ایپ پر بذریعہ میسیج رابطہ کریں اور جواب کا انتظار فرمائیں۔ برائے مہربانی اخلاقیات کا خیال رکھتے ہوئے موبائل پر کال یا ایم ایس کرنے کی کوشش ہر گز نہ کریں۔ ورنہ گروپس سے تو ریموو کیا ہی جائے گا بلاک بھی کیا جائے گا۔

نوٹ: ہمارے کسی گروپ کی کوئی فیس نہیں ہے۔ سب فی سبیل اللہ ہے

0306-7163117	0343-7008883	0333-8033313
محمد سلمان سلیم	پاکستان زندہ باد	راؤ ایاز

پاکستان زندہ باد　　　　پاکستان پائندہ باد

اللہ تبارک تعالیٰ ہم سب کا حامی وناصر ہو

2) Analyzing the Problem:

The programmer investigates the problem and gathers as much information as possible to find a solution. The following questions can be asked to analyze the problem.

- Is it possible to solve the problem on a computer?
- What is to be done to find the solution of the problem?
- What is the proper and what output is required?
- How many solutions are possible?
- Which solution is the best and why?
- How solution will be implemerted?

3) Planning The Solution Of The Problem:

It refers to dividing the solution of any problem into steps and arranging them into proper order.

How to Discuss a Problem:

- Talk to the right person.
- Focus on talking about the problem only.
- Stay calm and reasonable.
- Explain the problem in detail and provide any evidence you may have.
- Tell them what outcome you want.
- Listen to their response.
- Does his/her response resolve your issue?

4) Candid Solutions of a Problem:

All the possible solutions of a problem that produce correct result are known as candid solutions. To find candid solutions of a problem, programmer has to look for different methods to solve the problem and come up with several solutions.

5) Select The Best Solution:

After finding the candid solutions, only one solution can be selected. The selection of final solution of a problem shoulc be based on the followings:

- **Speed**: It means when the solution is implemented in a programming language, the program should run fast.
- **Cost:** The selected solution should be cost-effective.
- **Complexity:** The selected solution of the problem should not be complicated.

Write an algorithm to calculate the area of a rectangle for given breath and length?

Step 1 Input the width (W) and Length (L) of a rectangle
Step 2 Calculate the area (A) by multiplying L with W
Step 3 Print A

Write an algorithm that inputs length in inches and calculate and prints it in centimeters.

Step 1 Input the length in inches (LI)
Step 2 Calculate the length in cm (LCM) by multiplying LI with 2.54
Step 3 Print LCM

Write an algorithm that inputs marks and prints the message "PASS" or "FAIL". Passing marks are 33.

Step 1 Input Marks (M)
Step 2 Check if (M<33) then Print "FAIL" GOTO Step 4
Step 3 ELSE Print "PASS"
Step 4 Stop

Write an algorithm to find the sum of given sequence.

Step 1 Start
Initialize Sum to 0 and K=5
Step 2 Add K to SUM
SUM=SUM+K
Step 3 Increment K by 5
K=K+5
Step 4 Check if the value of K is less than or equal to 60
IF K≤100 THEN GOTO Step 2 otherwise GOTO Step 5
Step 5 Print SUM
Step 6 Stop

Write an algorithm to find the product of given numbers.
PRODUCT = 1×3×5×7×9×11×13×15

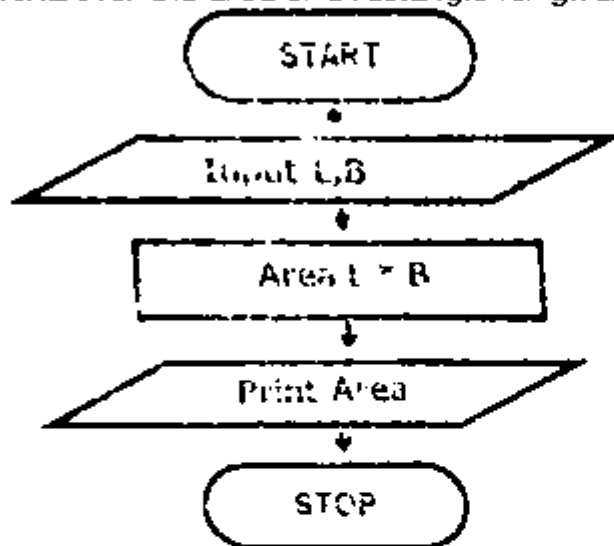
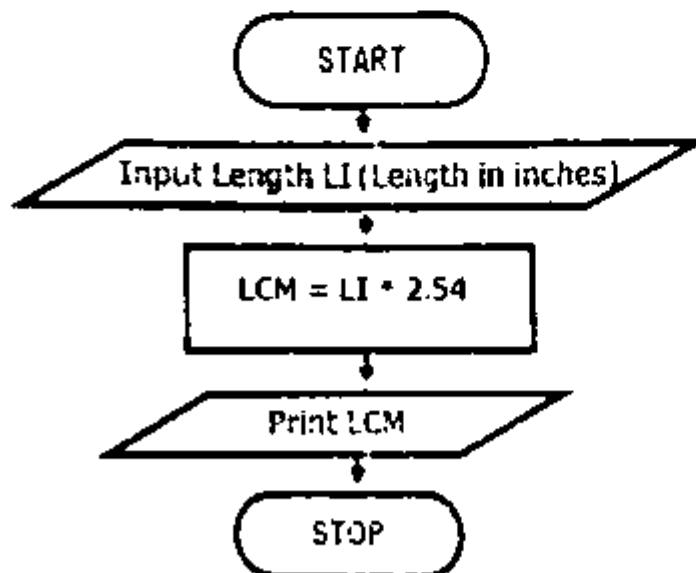
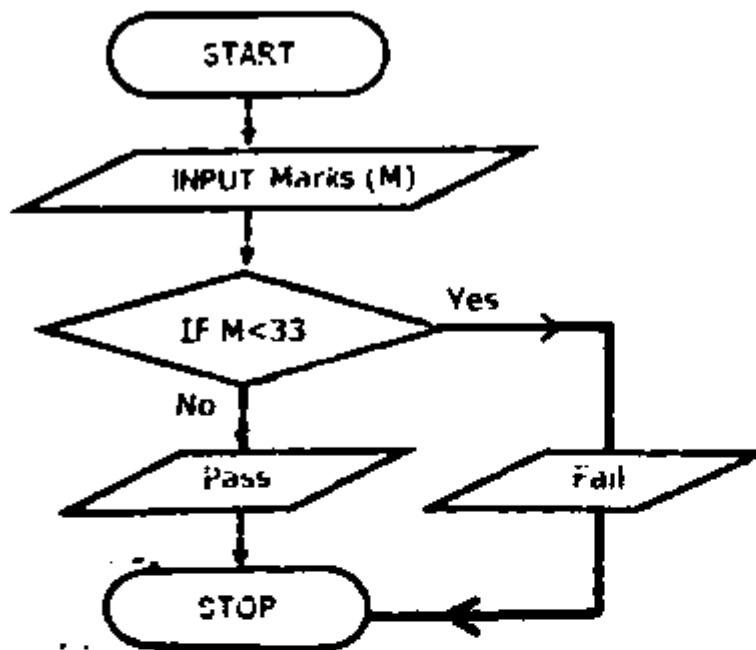
Step 1 Start
Initialize variable K to 1 and prod = 1
K=1 Prod= 1
Step 2 Increment K by 2
K=K+2
Step 3 Find the Product
Prod=Prod × K
Step 4 Check if the value of K is less than 16
IF K<16 THEN GOTO Step 2 otherwise GOTO Step 5
Step 5 Print product
Print Prod
Step 6 Stop

Write an algorithm to print multiplication table of a number in reverse order.

Step 1 Enter the number N whose table is to be generated
Step 2 Initialize the valve of I with 10
I=10
Step 3 Find the product of N and I
Prod= N × I
Step 4 Print N, I and Prod
Print N I and Prod

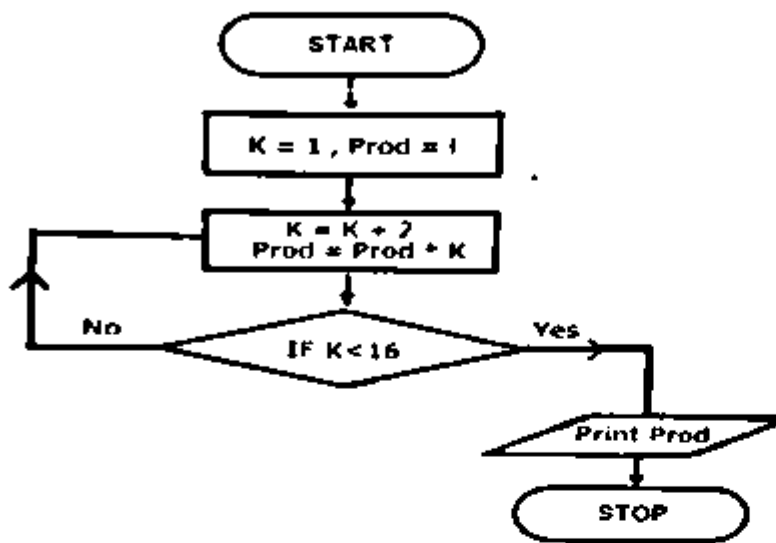
Step 5 Decrease the value of I by 1
I=I - 1
Step 5 IF the value of I is > 0 THEN GOTO Step 3 Otherwise GOTO Step 6
Step 6 Stop

Write a Flowchart to calculate the area of a rectangle for given breadth and length.

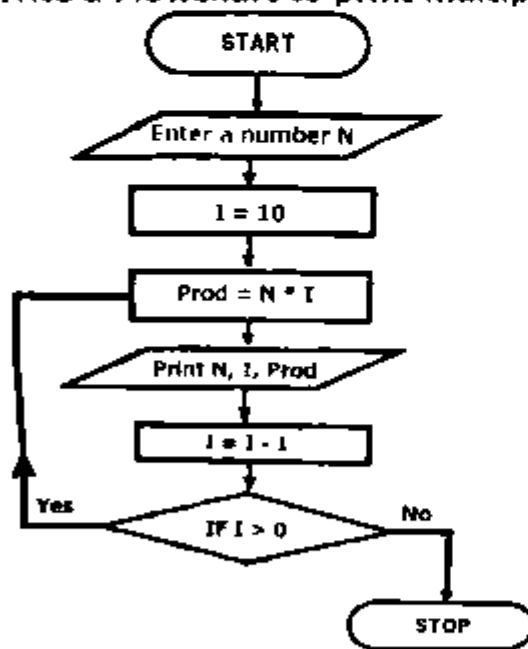
Flowchart for the area of a rectangle for given breadth and length:

Write a Flowchart to that inputs length in inches and calculate and prints it in centimeters.

Write a Flowchart to that inputs marks and prints the message "PASS" or "FAIL" Passing marks are 33.

Write a Flowchart to find the sum of given sequence.

SUM = 20+25+30+40+45+50+55+60

Write a Flowchart to to find the product of given numbers.

PRODUCT = 1×3×5×7×9×11×13×15

Write a Flowchart to print multiplication table of a number in reverse order.

Written/Composed by: - SHAHZAD IFTIKHAR Contact # 0313-5665666
Website: www.downloadclassnotes.com , E-mail: raoshahzadiftikhar@gmail.com

==

SHORT QUESTIONS

Define computer program.

- A computer program is a set of instructions (statements) written in a programming language to solve a problem.
- There are many programming languages for writing computer programs.
- Any task performed by a computer is controlled by a set of instructions that are executed by the microprocessor.

Differentiate between syntax and semantic.

Syntax:

- Syntax refers to the rules of a programming language.
- Statements are written as per syntax.
- It is just like the grammar of a natural language.
- For example, an assignment statement consists of a variable and an expression separated by equal sign as an assignment operator.
  Variable = expression;

Semantic:

- Semantic gives meaning to statements of a programming language.
- It describes the sequence of operations to be performed.
- For example, in the assignment statement;
  Sum = a + b;
- The semantic of the statement is to perform the expression, that is add the values stored in variables **a** and **b** and then store the result in variable **Sum**.

Write three differences between assembly language and HLLs.

Feature of high level language (HLLs):

- These are easily understandable.
- The programs written in HLL are portable.
- Debugging process is easy.
- Written program is not machine dependent.

Features of Assembly language:

- These are not easily understandable. However, they are relatively easier than machine level languages.
- The programs written in Assembly languages are not portable.
- Debugging process is not very easy.
- Written program is machine dependent.

Write four characteristics of HLLs.

- These languages were developed to make programming simple, easier and less prone to errors.
- High level languages are not machine dependent.
- The process of finding and removing errors in programs (debugging) is easier.
- High-level language programs are highly structured. They allow programmers to break lengthy programs into a number of modules.

==

===

Define Integrated Development Environment (IDE).

- IDE is computer software that brings all the processes and tools required for program development into one place.
- Its aim is to make the life of programmers easier by grouping together all the tasks needed for building applications into one environment.
- Modern IDEs has easy GUI interface.
- Most of the new programming languages use IDE to create, compile and run programs.

Differentiate between constant and variable.

Constants:

- Constants are quantities whose values do not change during program execution.
- The constants has following types:

  i. Numeric Constants:

  Numeric Constants are of two types, integer and floating-point numbers.

  - **Integer constants** represent values that are counted. For example, the numbers of students in a class. Some examples of integer constants are 9535, -724, 31 etc.
  - **Floating-point constants** are used to represent values that are measured. For example the height of a person is 166.75 cm or the weight such as 82.6 kilograms.

  ii. Character Constant:

  Character Constant is one of the symbols in C character set. It includes digits from 0 to 9, upper-case letters A to Z, lower-case letters a to z, punctuation symbols such as semicolon (;), comma (,), period (.) and special symbols such as +, -, =, > etc. A character constant is enclosed by single quotes such as 'a', 's', etc.

  iii. String Constant:

  String Constant contains a string of characters within double quotes such as "Hello Wahaj", "a", etc.

Variable:

- A variable is a symbolic name that represents a value that can change during execution of a program.
- A variable has a name, known as variable name and it holds data of other types.
- A number or any other type of data held in a variable is called its value.
- It has following types:

  i. Numeric variables:

  Numeric variables are used to represent numeric values in computer programs. They represent integer and floating-point values. Some examples of numeric variables are sum, average, length, salary, marks, etc.

  ii. Character variables:

  Character variables represent character values in computer programs. It can represent a single character or a string of characters. Some example of character variables are name, city, gender, etc.

===

Which of the following are valid C variables? Give the reason if not a valid variable area, 5x, Sum, net pay, float, _age, else, case, size22, my_weight?

Valid Variables:

- My_weight
- Area
- size22
- Sum

Not Valid Variables:

5x	A variable can't be start with number
Net pay	Space can't be used in variable name
Float	Data types can't be used in variable name
_age	Special character (_) can't be used in variable name
Else	Commands cannot be used as variable name
Case	Commands cannot be used as variable name

What are reserved words? Why they should not be used as variable names.

- The words that are part of programming language and have special purposes in computer programs are called reserved words or keywords.
- They have predefined use and cannot be used for any other purpose.
- These are always written in lowercase.
- There are 32 words defined as reserved words in C which are as under:

RESERVED WORDS

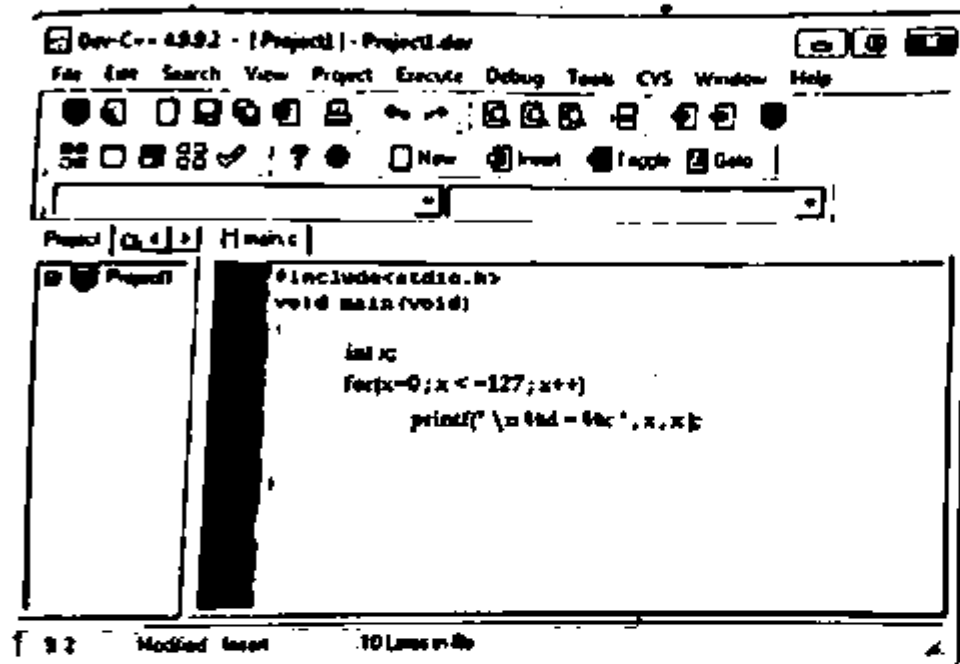
Auto	double	Int	Struct
Break	Else	Long	Switch
Case	Enum	Register	Teepee
Char	Extern	Return	Union
Const	Float	Short	Unsigned
Continue	For	Signed	Void
Default	Goto	Sizeof	Volatile
Do	If	Static	while

Why comments are used in programs?

- Adding Comments is a good programming practice.
- Comments makes the program understandable for others.
- Comments in the source code are ignored by the compiler.
- Generally, programmers write comments at the beginning of the program explaining thereunder what the program is intended to achieve.
- There are two types of comments
  - Single line comment
  - Multiple line comment

What is the use of typecasting in C programs?

- Typecasting is a method to convert a variable from one data type to another data type during program execution.
- For example, a variable of type **int** can act as a variable of type **char** using typecasting.

- For example, a one use for typecasting is when there is need to produce ACII characters for decimal codes (0-127). To do this, the programmer will need format specifier %c.
- There are two types of typecasting in C.
    - Implicit typecasting
    - Explicit typecasting

LONG QUESTIONS

Describe the following HLLs: C/C++, Visual Basic, C# and Java

C/C++:

- It was developed in early 1970s by Dennis Ritchie at Bell Laboratories.
- C has become one of the most popular programming languages today.
- It is a highly structured programming language which is easy to understand and use.
- In the past, it was mainly used for writing system programs such as operating systems, compilers, assemblers etc.
- Today it is used for writing all types of application programs. For example word-processing programs, spreadsheet programs, database management systems, educational programs, games etc.
- C++ was developed by Bjarne Stroustrup also at Bell Laboratories during 1983-1985.
- C++ is a superset of C, meaning that any valid C program is also a valid C++ program.
- The purpose of developing C++ was to provide programming facilities to easily and quickly write more powerful programs.

Visual Basic:

- Visual Basic (VB) is a high level language which evolved from the earlier version called BASIC.
- BASIC stands for Beginner's All-purpose Symbolic Instruction Code.
- VB is a very popular programming language for writing windows and Web applications.
- It provides a graphical development environment.
- VB is commonly used for developing business programs such as payroll system and inventory control program.
- The user can also write programs related with engineering, science, art, education, games etc.

C#:

- C# (C-sharp) is a language developed in 2000 by Microsoft Corporation.
- It is a simple, modern, general-purpose programming language.
- Syntax of C# is very similar to C and C++.
- It also has some features of Java.
- It is a language that makes computer programming easy and efficient.
- It provides facilities to write Web applications that can be used across the internet.
- All types of programs including games, utilities, operating system, compilers, business applications and Web based applications can be developed in C#.

Java:

- Java is a high-level language developed by Sun Microsystem.
- It is very similar in syntax to C and C++.
- Java is an ideal language for network computing.
- It is used for writing programs for many devices, computers and networks.
- It is widely used in web applications.
- The current versions of most of the Web browsers are made Java enabled.

Differentiate between compiler and interpreter.

Compiler:

- A compiler is computer software that translates source program into object program.
- Source program consists of statements written in a high level language such as C, Pascal and Java etc.
- For example, a program written in C language by a programmer to print table of a number is known as **source program**. When it is translated with a compiler into machine language, the resulting program is known as **object program**.
- **The object program** is understandable by computer processor but difficult for a human to read and understand because it consists of zeroes and ones.

Interpreter:

- Interpreter translates high level language programs into machine language but it translates one instruction at a time and executes it immediately before translating the next instruction.
- Examples of programming languages that use interpreter are Java script, BASIC, Visual Basic and Perl.

- Interpreter reads each statement of source program, one at a time and determines what it means as it executes it.
- It means each time a statement is read, it must be translated into machine language before execution.

Describe the functions of linker and loader programs.

Linker:

- Linker is software that translates object program into a single executable program.
- During this process, if it could not find the definition of a particular function that is used in the program, then it would assume that it is definec in C library.
- It will replace this function in the object program with the code from C library and then create a single executable program.

Loader:

- It is software that loads programs into memory and then executes them.

What are the rules for specifying a variable name in C language?

- A variable begins with a letter or underscore (_) and may consist of letters, underscores and or digits.
- The underscores may be used to improve readability of the variable name. For example, **over_times**.
- There is no restriction on the length of a variable name. However, only the first 31 characters of a variable are significant.
- Both upper and lower case letters are allowed in naming variables. An upper case letter is considered different from a lower caser letter, for example the variable AVG is different from Avg or avg.
- Special characters cannot be used as variable name. e.g. #, ? , @ etc.
- Reserved words of C language such as **int, case, if,** etc. cannot be used as variable names.
- There must be no embedded blank in the name of variable, for example (**mass ss**) is not correct.

What is the difference between implicit type casting and explicit type casting? Give examples.

- Typecasting is a method to convert a variable from one data type to another data type during program execution.
- For example, a variable of type **int** can act as a variable of type **char** using typecasting.
- For example, a one use for typecasting is when there is need to produce ACII characters for decimal codes (0-127). To do this, the programmer will need format specifier %c.
- There are two types of typecasting in C.

Implicit Typecasting:

- Implicit typecasting is performed automatically by the compiler without programmer's intervention.
- In this type of casting the compiler converts all operation into the data type of the largest operand.

==

- Suppose an expression contains variables of integer and double data types, the result will be evaluated to a double data type.

For example in the program given below, compiler converts the sum of two float numbers to an **int**.

```
#include<stdio.h>
void main(void)
{
  float value1 = 2.5;
  float value2 = 5.3;
  int result;
   result = value1 + value2;
   printf("Result : %d", result);
}
```

Explicit Typecasting:

- Explicit casting is performed by programmer.
- The programmer explicitly defines the data type in parenthesis before the variable or expression, as follows:
  (type) expression

For example in the program given below, the programmer converts division of two int numbers to float.

```
#include <stdio.h>
Void main(void)
{
 Int value1 = 30;
 It value2 = 7;
 Float result;
 Result = (float)value1 / value2;
 Printf("Result : %f", result);
}
```

What is a Preprocessor Directive? Give Examples.

- Preprocessor directive are instructions for the C compiler.
- Every language program contains certain preprocessor directives at the beginning of the program.
- Before translating a C language program into machine language, the compiler of C language carries out the processor directives.
- These directives start with number sign (#)
- The most commonly used preprocessor directives are **include and define.**

The include Preprocessor Directive:

- It has the following syntax.
  #include<header file name>
- When this preprocessor is carried out by the C compiler, it will search for the header file that is written within the less than (<) and greater than (>) symbols and copy it into the source file.

==

- For example when we include **stdio.h** in any program, it tells the C compiler to copy the **stdio.h** header file into the program. The **stdio.h** header file stand for standard input-output header. It includes the standard **prindf()** and **scanf()** function prototypes. In the above program the **printf()** function is used. Therefore, it is required to include this header file in the include preprocessor directive.

The define Preprocessor Directive:

- The define preprocessor is used for defining constants (i.e. Symbolic constants) in C programs.
- It directs the compiler to replace all the occurrences of using this directive are to give a meaningful name to a constant value that is to be used in the program.
- It has the following syntax.
  #define SYMBOL value/expression
- Here, SYMBOL is a valid C variable name and it is by convention written in uppercase. An examples of define preprocessor directive is given below.
  #define HEIGHT 12
- This preprocessor directive tells the compiler to replace all the occurrences of the variable **HEIGHT** with the value 12, in the program during its execution.
  #define CUBE(a) (a*a*a)
- It can be used in a program as given below.
  volume=CUBE(side);
- The volume will be calculated as:
  volume=(side*side*side);

Written/Composed by: - SHAHZAD IFTIKHAR Contact # 0313-5665666
Website: www.downloadclassnotes.com , E-mail: raoshahzadiftikhar@gmail.com

===

SHORT QUESTIONS

Why format specifier is used? Explain with examples.
A format specifier is computer code that tells about the data type, field, width and the format according to which a value is to be printed or read from an input device. Following are the commonly used format specifiers:

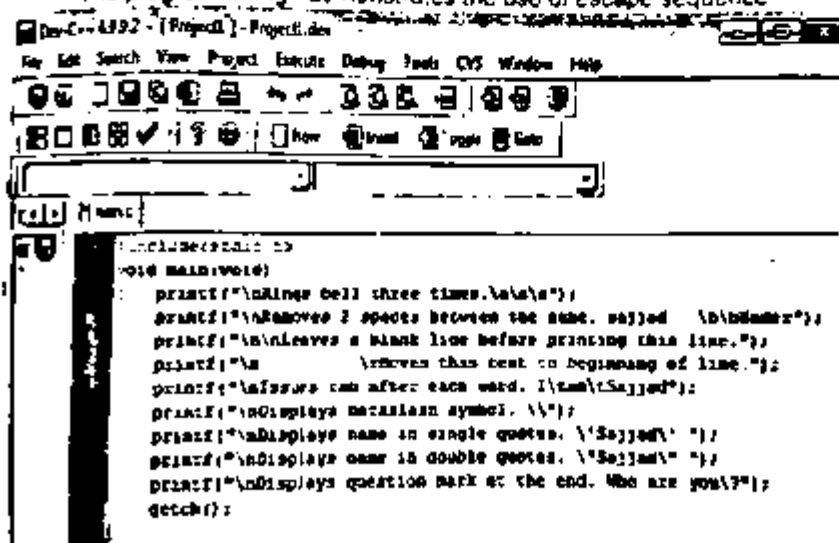
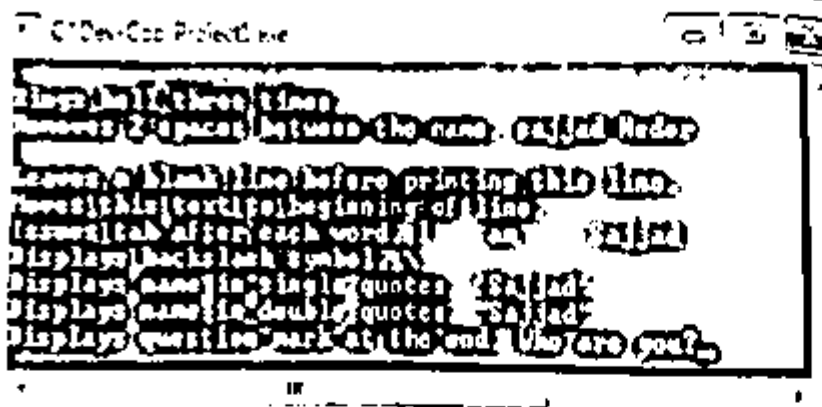
%d = Decimal Integer
%i = integer
%ld = long decimal integer
%f = floating-point (decimal notation)
%g = floating-point (exponential notation)
%e = floating-point (%f or %g, whichever is shorter)
%c = single character
%s = string

Why escape sequence is used? Explain with examples.
The special characters used in C language to control printing on the output device are called escape sequences. These characters are not printed. An escape sequence is a combination of a backslash (\) and a code character. The backslash is called the control character. A list of commonly used escape sequences is given below:

Escape Sequence	Meaning
\a	Produces alert (bell) sound
\b	Moves cursor backward by one position
\n	Moves cursor to the beginning of next line
\r	Moves cursor to the beginning of current line
\t	Moves cursor to the next horizontal tabular position
\\	Produces a backslash
\'	Produces a single quote
\"	Produces a double quote
\?	Produces a question mark

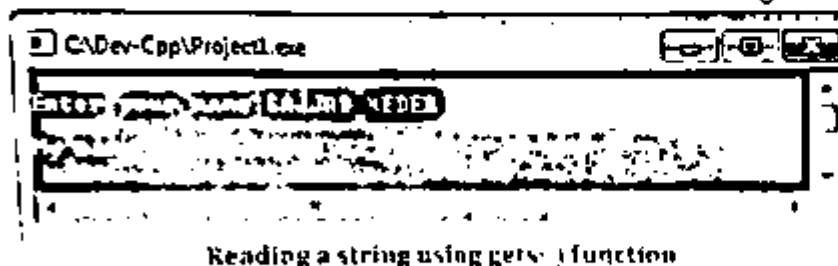
The program in Fig demonstrates the use of escape sequence

```
#include<stdio.h>
void main(void)
{
    printf("\nRings bell three times.\a\a\a");
    printf("\nRemoves 2 spaces between the name. Sajjad   \b\bHaider");
    printf("\n\nLeaves a blank line before printing this line.");
    printf("\n          \rMoves this text to beginning of line.");
    printf("\nInsers tab after each word. I\tam\tSajjad");
    printf("\nDisplays backslash symbol. \\");
    printf("\nDisplays name in single quotes. \'Sajjad\' ");
    printf("\nDisplays name in double quotes. \"Sajjad\" ");
    printf("\nDisplays question mark at the end. Who are you\?");
    getch();
```

===

The output of the program is shown in Fig.

What is the purpose of gets() function? Explain with an example.

The gets() functions is used to read a string from the keyboard and store it in the variable specified inside the parenthesis. The program given below demonstrates the use of gets() function in a program.

```
#include<stdio.h>
void main(void)
{
  char name[20];
  printf("\nEnter your name:");
  gets(name);
  printf("\nMy name is %s",name);
  getch();
}
```

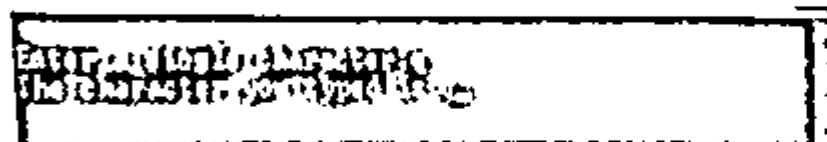
In this program, the format specifier %s is used to print the string stored in the variable name. The execution of this is shown in Fig.

Reading a string using gets( ) function

Differentiate between getch() and getche() functions.

- Sometimes in programming, it is required to read a single character as soon as it is typed, without waiting for Enter key to be pressed.
- For example, in a game the user might want an object to move each time he presses one of the arrow keys. It would be awkward to press the Enter key each time the user presses and arrow key.
- The getche() function is used for this purpose. The get means it gets something from an input device. The 'ch' means it gets a character and the 'e' means it echoes (displays) the reads a single character to the screen when it is typed.

- The program in Fig below reads a single character the instant it is typed and displays it on the screen. The user does not have to press the Enter key after typing the letter.

```
#include <stdio.h>
#include <conio.h>
void main(void)
{
  char ch;
  printf("\nEnter a single character:");
  ch=getche();
  printf("\nThe character you typed is %c",ch);
  getch();
}
```

- In this program, the getche() function reads a single character from the keyboard and it is assigned to the variable ch. The format specifier %c is used to print the letter stored in variable ch. The execution of this program is shown in Fig.

- The getch() is a similar function to getche(), the difference is that it does not display the typed character t other screen. The getche() and getch() functions require conio.h header file, so it must be included in the program as well.

Evaluate the following expressions.

7+5*(3+4)	**100/10/4**	**50%13%3**	**30/7*3-6**
= 7+5*7	= 10/4	= 11%3	= 4.28*3-6
= 7+35	= 2.5	= 2	= 12.8-6
= 42			= 6.8

What will be the output of the following program?

```
#include <stdio.h>
Void main(void)
{
      int x,y,z1,z2,z4,z4
      Y=5;
      y=5;
      printf("\nz1=1=%d",z1)'
      z2=x%y'
      z3=++z'
      printf("\nz2=%d",z2)'
      z3=++z'
      printf("\nz4=%d",z4);
}
```

Ans: Output:

Z1 = 3
Z2 = 2
Z3 = 18
Z4 = 5

What will be the output of the following Program?

```
#include<stdio.h>
Void main(void)
{
	int b;
	floata,c,d,e,f;
	a=1484;
	b=7;
	c=a-b;
	printf("\nc=%f",c);
	d=a/b;
	printf("\nd=%f",d);
	e=a-b3;
	printf("\ne=%f",e);
	f=(a+b)/2;
	printf("\nf=%f",f);
}
```

Ans: Output:

C= 7.840000
D = 2.120000
E = 6.160000
F = 10.920000

LONG QUESTION

Describe how basic and compound assignment operators are used?

- They are used to assign values to variables used in computer programs.
- C language provides three types of assignment operators.
- These are:
  - Basic assignments operator
  - Compound assignment operators
  - Increment/decrement operators.

Basic Assignment Operator:

- The basic assignment operator is =.
- This is used to assign value of an expression to a variable.
- It has the general form:

Variable = expression

- Where expressions may be a constant, another variable to which a value has previously been assigned or a formula to be evaluated. For example:

  Sum = a + b;

Compound Assignment Operators:

- In addition to =, there are a number of assignment operators unique to C.
- These include +=, -=, *=, /= and %=.
- Suppose **op** represents an arithmetic to assign value of an expression to a variable.

  Variable op = expression

- This is equivalent to:

  Variable = variable op expression

- For example, consider the following statement:

  Sum = sum + n;

- This assignment statement could be written using a compound assignment operator as:

  Sum += n;

- The effect is exactly the same but the expression is more compact. Some more examples are:

Sum -= n	is equivalent to	sum = sum – n
Prod*= n	is equivalent to	prod = prod * n
A /= b	is equivalent to	a = a / b
A %= b	is equivalent to	a = a % b

Describe the functions of the following operators?

i) Relational operators

ii) Logical operators

iii) Conditional operator

A) Relational Operators:

- These are used to compare two values of the same type.
- These are used in expressions when a decision is to be based on a condition.
- After evaluation of a relational expression, the result produced is either True of False.
- Relational operators are used in programming for decision making.
- There are six types of relational operators in C language. These are described in table below:

Operator	Definition

= =	Equal to
!=	Not equal to
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

Following are some examples of relational operators:

c>=a+b
x<5.3
n==20
count!=10

- If **a** has the value 25, **b** has the value 10 and **c** the value 28 then first expression is false since 28 is not greater than or equal to 35.
- In the second expression, if x has the value 4.5, the expression is true because x is less than 5.3.
- In the last expression if court is any number other than 10 then the expression will be true. It will only be false when count is equal to 10.

B) Logical operators:

- Logical operators are used for building compound conditions.
- A single condition is built using a relational operator in an expression.
- If we need to build more than one condition for some action to take place programming, then we have to form compound condition.
- They have following types:

Operator	Definition
&&	AND
||	OR
!	NOT

Logical AND (&&) Operator:

- It is used to form compound condition in which two relational expressions are evaluated.
- One relational expression is to the left and the other to the right of the compound condition is considered true otherwise it is false.

Syntax

Expression1 && Expression2

Truth table for AND operator is shown here under.

Expression-1	Expression-2	Expression-1 && Expression-2
True	True	True
True	False	False

False	True	False
False	False	False

Example:
Consider the following compound condition:
(a>=1)&&(a<=10)

This expression is considered true if both conditions (a>=1) and (a<=1) are true, that is, if the value of **a** is between 1 to 10.

In the next example, the compound condition is considered true if the value stored in **a** is greater than the value stored in **b** and the value stored in **c** is equal to 15.
(a>b)&&(c==15)

The following compound condition will check whether the character stored in variable **ch** is a lowercase letter or not.
(ch>='a')&&(ch<-'z')

Logical OR (||) Operator:
Logical **OR** operator is also used to form a compound condition just like the logical AND operator, one relational expression is to the left and the other to the right of the OR operator.

The compound condition is true if either of the conditions is true or both conditions are true. It is considered false only if both of the conditions are false.

Syntax:
Expression1|| Expression2

Truth table for AND operator is shown here under:

Expression-1	Expression-2	Expression-1 && Expression-2
True	True	True
True	False	True
False	True	True
False	False	False

Example:
(n<10)||(n>25)
Suppose, the value of **n** is 5, then the expression will be considered true because one of the two conditions is true if the value of **n** is 28 then also the compound condition will be true. If the value of **n** is 12 then the expression will be false since both conditions are false.

The next compound conditions will be true if **a** is greater **b** or **c** is equal to 10. It will also be true if both conditions are true, that is, **a** is greater than **b** and **c** is not equal to 10.

(a>b) || (c==10)

Logical OR condition is used when we wish to perform an operations if one of the two conditions is true or both of the conditions are true.

Logical NOT (!) Operator:

The logical NOT operator is used with a single expression (condition) and evaluates to true if the expression is false and evaluates to false if the expression is true. In other words, it reverses the result of a single expression.

Syntax:

!Expression1

Truth table for AND operator is shown here under;

Expression	**!Expression**
True	False
False	True

For example, the expression:

!(a<b)

C) Conditional Operator/ Conditional (Ternary) Operator:

A conditional operator is a decision-making operator. It has the following form.

Condtion?expression1: expression2:

When this statement is executed, the condition is evaluated. If it is true, the entire conditional expression takes on the value of expression1. If it is false, the expression takes on a value and can therefore be used in assignment statements. Consider the following example.

A= (k>15)? x*y : x+y;

This statement will assign the product of x and y to the variable **a**, if **k** is greater than 15, otherwise **a** will be assigned the sum of x and y. this expression is equivalent to the following if-else statement which will be explained in the next unit.

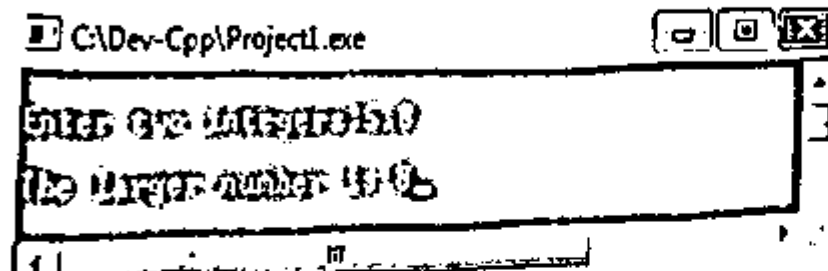
If (k>15)
A = x*y;
Else
A = x+y;

Some programmers may prefer to use the above if-else statement rather than using the conditional operator because it is easy to understand.

The program in Fig, demonstrate the use of conditional operator for finding the larger of two numbers.

```
#include <stdio.h>
#include <conio.h>
void main(void)

    int a, b, larger;
    printf("\nEnter two integers:");
    scanf("%d,%d", &a, &b);
    larger = a>b? a : b;
    printf("\nThe larger number is %d", larger);
   getch();
}
```

The execution of the program s shown in Fig.

Write a program that reads three numbers and prints their sum, product and average.

```
#include <stdio.h>
int maint()
{
      int num1, num2, num3, sum, prod, avg;
      printf("Enter the three numbers separated by ',' \n");
      scanf("%d, %d, %d, &num1, &num2, &num3);
      sum = num1 + num2 + num3;
      prod = num1 * num2 * num3;
      avg = sum/3;
      prinf("the sum of the three numbers is : %d\n",sum);
      printf("the prod of the three numbers is : %d\n",prod);
      printf("the average of the three numbers is : %d\n", avg);
}
```

Write a program that reads the length and width of a rectangle and prints its area.

```
#include <stdio.h>
main()
{
      float length, width, area;
      printf("Enter length of rectangle:");
      scanf("%f", &length);
      printf("Enter width of rectangle:");
```

```
    scanf("%f", &width);
    area= length*width;
    printf("Area of rectangle - %f sq. units ",area);
}
```

Write a program that reads the length of one side of a cube and prints its volume.

```
#include <stdio.h>
#include <conio.h>
main()
{
    float side, volume;
    printf("Enter length of any side of cube\n");
    scanf("%f", &side);
    printf("Volume of Cube: %0.4f\n, volume);
}
```

Write a program that reads temperature in Celsius, Converts it into Fahrenheit and prints on the screen.

```
#include <stdio.h>
#include <conio.h>
main()
{
    float fahren, celsius;
    printf("Enter the temperature in celsius\n");
    scanf("%f", &celsius);
    fahren =(9.0/5.0)*celsius + 32;
    prinft("%.2fC is equal to %.2fF\n:, celsius, fahren);
}
```

Write a program that reads name and address of a person and prints it on the screen using gets() and puts() functions.

```
#include <stdio.h>
#include <string.h>
int main()
{
    char name[50], add[50];
    printf("Enter your name:");
    gets(name);
    printf("Enter your address:");
    gets(add);
    printf("Your name is:");
    puts(name);
    printf("Your address is:");
    puts(add);
}
```

==

SHORT QUESTIONS

Differentiate between if condition and if else condition.

If Selection Structure:

The if statement has the following general form / Syntax.

```
if (conditions)
{
	Block of statements
}
```

If-else Selection Structure:

The if-else statement is used in situation where some code is to be executed if a condition is true and some other code is to be executed if the condition is false.

The if-else statement has the following general form / Syntax.

```
if (condition)
{
	Block of statements
}
else
{
	Block of statements
}
```

Differentiate between else-if and switch selection structures.

Else-if Selection structures:

The **if-else** statement is used in situation where some code is to be executed if a condition is true and some other code is to be executed if the condition is false.

The if-else statement has the following general form / Syntax.

```
if (condition)
{
	Block of statement
}
else
{
	Block of statement
}
```

When if-else statement is executed, the condition is evaluated.

Switch Selection Structures:

The switch statement has the following general form

```
Switch (expression)
{
case const-1:
	statements;
	break;
case const-2:
	statements;
```

==

```
            break;
            .
            .
            .
        default:
            statements;
        }
```

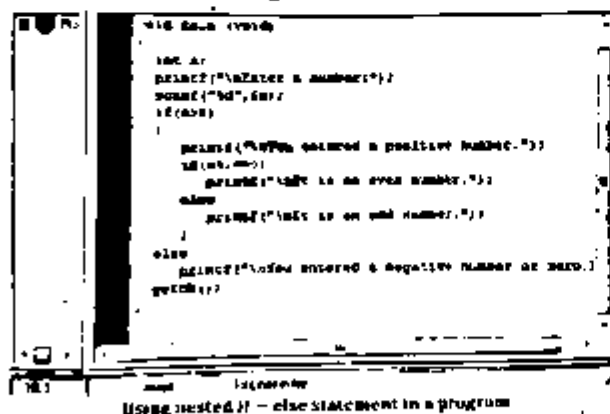
The switch statement is similar to the else-if statement. It is used when multiple choices are given and one choice is to be selected.

What is nested selection structure?

The selection structure that is within another selection structure is known as nested selection structure. This is also supported in C language in C language; the programmer can have a selection structure **(if, if-else, else-if** or **switch statement)** within another selection structure.

Program:

- When this program is executed, it reads an integer number, stores it in the variable **n** and then the condition (n>0) is evaluated.
- If it is true, it prints the message that the entered number is a positive number and then the nested if-else statement is executed.
- The nested if-else statement prints whether **n** is an even number or an odd number.
- If the condition (n>0) is false then the statements following the first **if** are skipped and the last statement after the else is executed which prints the message that the user entered a negative number or zero.

Using nested if – else statement in a program

Write the following statement using if-else statement.

K = (a+b>20)? A+3*b; a-b;=

```
if (a+b)>20
      K = a+3*b;
else
      K = a-b;
```

Write the following statement using conditional operator.

```
If (x>y)
      z=(x+y)/3;
else
      z=x-5*y;
```

Ans: z=(x>y) ? (x+y)/3 : x-5*y;

What will be the output of the following code?

```
Int n, count=15; sum=30;
If (n<25)
{     count=count+5;
      printf("\nCount=%d",count);
      printf("\nSum=%d",sum);        }
else
{     count=count-5;
      Sum=sum+n;
      printf("\nCount=%d",count);
      printf("\nSum=%d",sum);        }
```

Ans: Count = 10
Sum = 58

What will be the output of the following code?

```
charch;
ch='c';
switch(ch)
{     case 'a':
       Printf("\nGood Morning! "); break;
      Case 'b':
      printf("\nHave a Nice Day! "); break;
      case 'c':
      case 'd':
      case 'e':
       printf("\n Good Bye! "); break;
}
```

Ans: Good Bye!

==

LONG QUESTION

What is control structure? Explain conditional control structure with examples.

Control Structure:

In a programming language, a control statement is an instruction which determines the sequence of execution of other statements in a program.

Conditional control structure:

Conditional Statement:

A conditional statement is an instruction in a programming language that contains a condition. When a conditional statement is executed, first the condition is evaluated and then based on the result (true or false) a particular statement or a set of statements is executed.

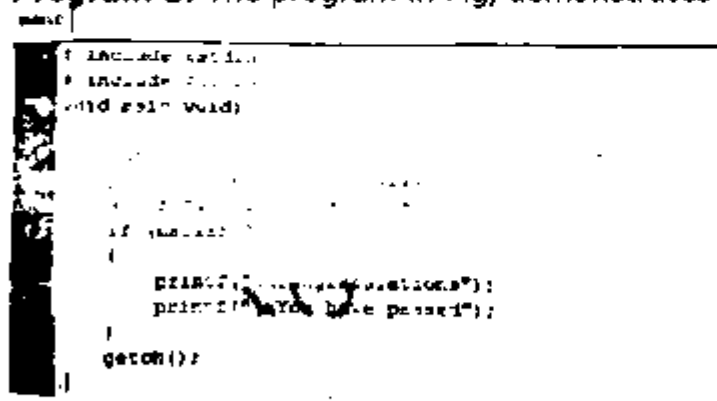
Conditional statements of C language are if, if-else, else-if and switch statements.

Structure of IF Statement:

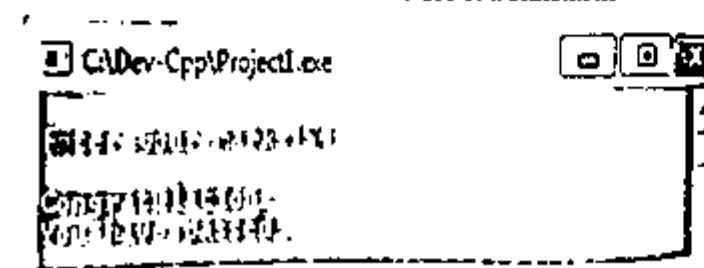
The **if** statement has the following general form.

```
    if (conditions)
{
    Block of statements
}
```

Use of If Statement:

Program 1: The program in Fig, demonstrates the use of **if** statement.

Program to demonstrate the use of if statement

Output of the program 1

==

Structure Of If-Else Statement:

The **if-else** statement is used in situation hwre some code is to be executed if a condition is true and some other code is to be executed if the condition is false.
The **if-else** statement has the fo lowing general form.

```
if (condition)
{
        Block of statements
}
else
{
        Block of statements
}
```

- When **if-else** statement is executed. The condition is evaluated.
- If the condition is true then the block of statements following **if** will be executed and the block of statements following **else** will be skipped.
- If the condition is false then the block of statements follcwing **if** will be skipped and the block of statements following **else** will be executed.
- If a single statement is to be executed after **if** or **else** then braces are not required.

Structure OF If-Else-If Statement:

The **else-if** is a type of conditional statement that combines more than two conditions. It allows the programmer to make a decision basec on several conditions.
The **else-if** statement has the following general form.

```
if(condition-1)
{
        Block of statements
}
Else if(condition-2)
{
        Block of statements
}
Else if(condition-3)
{
        Block of statements
        .
        .
        .
}
Else
{
        Block of statements to be executed
        When none of the conditions is true,
}
```

- When this statement is executed, contion-1 is evaluated, if it is true then the block of statements following **if** is executed and if it is false, the next condition is evaluated.
- If any condition is true then the following block of statements is executed.
- If none of the conditions is true then the block of statements following **else** is executed automatically.
- If a single statement is to be executed after **if, else-if** or **else,** instead of set of statement then the braces are not required.

Switch Statement:

The **switch** statement has the following general form.

```
Switch (expression)
{
Case const-1;
      Statement;
      Break;
Case conts-2;
      Statements;
      Break;
      .
      .
      .
Default:
      Statements;
}
```

- The **switch** statement is similar to the **else-if** statement. It is used when multiple choices are given and one choice is to be selected.

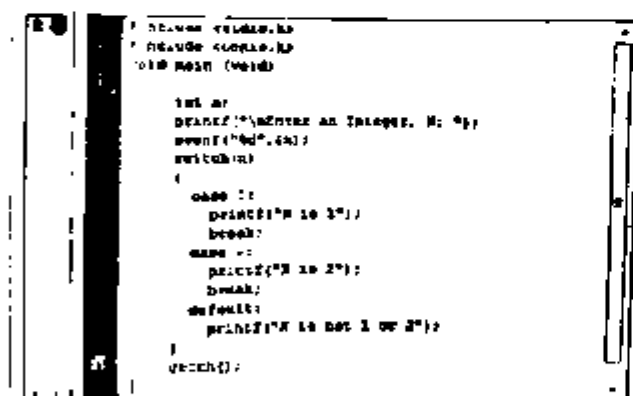
What is the purpose of switch ( ) statement? Explain with help of example.

Switch Statement:

The **switch** statement has the following general form

```
Switch (expression)
{
Case const-1;
      Statement;
      Break;
Case conts-2;
      Statements;
      Break;
      .
      .
      .
Default:
      Statements;
}
```

Purpose of switch statement:

- The **switch** statement is similar to the **else-if** statement. It is used when multiple choices are given and once choice is to be selected.
- When **switch** statement is executed, the expression is evaluated. Based on the result of expression one of the cases in the **switch** statement is executed. The result of expression is compared with the constant values given after the key word **case**. If the result matches the constant value after any **case** then the statements under that case are executed.
- In switch statement, it is allowed to use a variable within the parenthesis instead of an expression based on which statements under a **case** can be executed.
- The purpose of **break** statement is to exit the body of the **switch** statement after executing the statements under a **case** and transfer control to the first statement following the end of the **switch** statement.
- If no **case** is matched then the statements under the **default** keyword are executed. Its use is optional if it is not used then the control exits from the body of the **switch** statement and goes to the first statement following the end of the **switch** statement.
- The expression should be of type **int, char** but not **float.**

- When this program is executed, the switch variable must have an integer value. The value of switch variable **n** is compared with the constant values statements following that particular **case.**
- If the switch variable does not match any of the **case** constants, control goes the keyword **default** which is at the end of the switch statement.
- Notice the use of **break** statement in this program. It terminates the **switch** statement when the body of the statements in a particular **case** has been executed.

SHORT QUESTION

Differentiate between for loop and while loop.

For loop:

The **for** statement is used to execute a set of statements repeatedly for a fixed number of times in a program.

While loop:

The **while** statement is used to implement repetition when the number of repetitions are not known in advance. In this case, repetition continues until some condition remains true.

Differentiate between while loop and do while loop.

WHILE() loop	DO-WHILE() loop
While() loop is pre-tested loop.	Do-While loop is post-tested loop.

The Syntax or general form of while() loop is:
While(condition)
{
Statements; //body of loop
}
	The Syntax or general form of do-while() loop is:
	Do{
	Statements; //body of loop
	}
	while(condition)

In 'while' loop the controlling condition appears at the start of the loop.	In 'do-while' loop the controlling condition appears at the end of the loop.
The iterations do not occur if, condition at the first iteration appears false.	The iteration occurs at least once even if the condition is false at the first iteration.

What will be the output of the following code?

```
int k;
for(k=1;k<=5;k++)
    printf("\n I am a student");
printf("\n GOOD BYE");
```

I am student
I am student
I am student
I am student
I am student
GOOD BYE

What will be the output of the following code?

```
int n;
for(n=30;n>=10;n=m-5)
    printf("\n%d",n);
```

30
25
20
15
10

Find errors in the following code.

```
Int k, a
A=3;
K=1;
while(k<10);
{
      printf("\n%f\t%f",k,k*a-1);
      k=k+2;
}
```

```
int k, a
a=3;
k=1;
while(k<10)
{
      printf("\n%d\t%d",k,k*a-1);
      k=k+2;
}
```

Output:

```
1   2
3   8
5   14
7   20
9   26
```

Convert the following for loop into a while loop.

```
int k;
for(k=25; K>0; k=k-3)
      printf("\n%d",k);
```

```
int k;
k=25;
while(k>0)
{
      printf("\n%d"d,k);
      k=k-3;
}
```

LONG QUESTIONS

What is looping structure? Explain for loop with examples.

Loop Structure:

A loop is a structure that enables the programmer to execute the same sequence of statements repeatedly until a particular condition is met.

For loop / For Statement:

- The **for** is a looping statement which is used to execute a set of statements repeatedly for a fixed number of times. It is also known as **counter loop**. It has the general form:

```
for (initialization; test condition; increment/decrement)
{
body of the loop
}
```

- When **for** statement is executed, a variable (also known as **loop variable**) is assigned an initial value in the initialization part of the loop, such as k=1 or count=0. The value of the loop variable is checked with the given test condition. The test condition is relational expression, such as k<10. If the condition is true, the control enters the body of the loop otherwise it will exit the loop.
- After the execution of the body of the loop, the control is transferred back to the increment/decrement part of the loop. The loop variable is incremented or decremented using an assignment statement such as k = k+1. The new value of loop variable is again checked with the test condition. If the condition is satisfied then the body of the loop is again executed. This process goes on till the test condition becomes false.
- Body of the loop may have one or more statements. If it contains only a single statement then braces are not needed.

Note: All the three expressions such an initialization, test condition and increment are optional. You can omit any or all in **for** statement.

Example: For example the following all for statement are valid.

```
For(; ;)
For(int i=1 ; ;)
For(I k<10 ; k++)
For(; ; k++)
For(: x<12 ;)
```

Explain while and do-while loops with examples.

While loops/ The While Statement:

A repetition structure when the number of iterations is not known in advance and the repetition continues until test condition remains true.

The **while** statement has the **general form:**

```
while (test condition)
{
      Body of the loop
}
```

- When a **while** statement is executed, the computer first evaluates the test condition. If it is true, body of the while loop is executed. After the execution of the body of the loop, the test condition is again evaluated and if it is true, the body of the loop is executed once again. This process continues until the test condition statement following the end of the body of loop.
- The body of the loop can be a single statement or it can be multiple statements. If the body of the loop consists of a single statement then the braces are not required but if it consists of more than one statement then braces must be used.

Examples:

Program: The program in Fig. prints all the upper-case letters are in the range 65 to 90.

```
#include<stdio.h>
#include<conio.h>
void main(void)
{
    int n;
    printf("\n");
    n=65;
    while(n<=90)
    {
        printf("%c ",n);
        n=n+1;
    }
    getch();
}
```

Output of the program is shown in Fig.

Output of Program

Do-while loops / The Do While Statement:

- The do while statement is used to implement loop structure when it is required to execute the loop at least once. The general form of the **do while** loop is given below.

```
do
{
    Body of the loop
}
While (test condition);
```

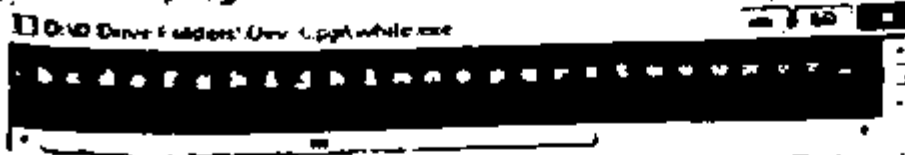
- The statement **while (test condition)** is placed at the end of the loop so that the body of the loop is executed at least once whether the condition is true or false. There is a semicolon after the test condition because it is at the end of loop if the body of loop contains a single statement then braces are not required.

Program:

The program in Fig. prints all the lower-case letters of the alphabet using **do while** loop. The ASCII codes for the lower-case letters are in the range of 97 to 122.

```
#include<stdio.h>
#include<conio.h>
void main(void)
{   int n;
    printf("\n");
    n=97;
    do
    {
        printf("%c  ",n);
        n=n+1;
    }
    while(n<=122);
    getch();
}
```

==

The output of this program is shown in Fig.

Output of Program

- In this program, the integer variable **n** is initialized to 97 which is the ASCII value of letter **a**, before entering the loop.
- The expression in the **do while** statement, **n<=122** acts as the test condition. The statement within the loop execute as long as this expression remains true.
- The statement **n=n+1** increments the value of **n** by one with each iteration. Note that the ASCII value of **z** is 122.

Explain the purpose of Break and continue statements with one example each.

The Break Statement:

C language provides the break statement to exit from a loop as soon as certain condition occurs. It is used in **for, while** and **do while** statements. Break statement is also used to exit the body of **switch** statement after executing the statements under a **case** and transfers control to the first statement following the end of the **switch** statement.

The Continue Statement:

Sometimes during the execution of a loop, when a certain condition occurs after executing a statement, it may be required to skip the remaining statements within the body of the loop and continue loop for next iteration until the test condition in met. C language provides the **continue** statement to achieve this task. The **continue** statement causes the loop to be continued with the next iteration after skipping the remaining statements within the body of the loop.

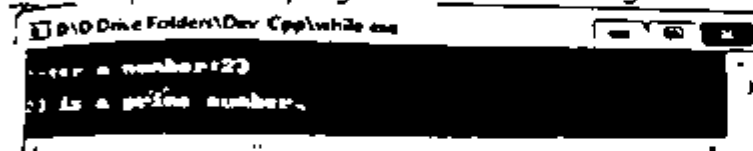
The general format of **continue** statement is:

Program to check for prime number by using break statement:

```
#include<stdio.h>
#include<conio.h>
void main(void)
{
    int num, i, a=0;
    printf("\nEnter a number:");
    scanf("%d",&num);
    for(i=2;i<=num/2;i++)
        if(num%i==0)
        {
            a=1;
            break;
        }
    if(a==0)
        printf("\n%d is a prime number.",num);
    else
        printf("\n%d is not a prime number.",num);
    getch();
}
```

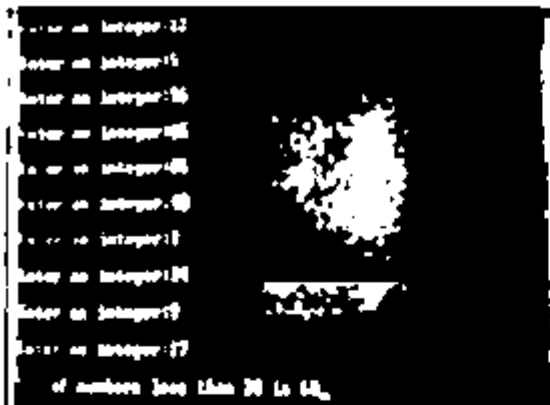
==

- When the above program is executed, it reads a number and determines whether it is a prime number or not. The integer variable a is assigned the value **0** to assume that the number **num** is a prime number.
- The remainder operator (%) is used inside the **for** loop to check if the number is exactly divisible by any number in the range **2** to **num/2**.
- If the number is exactly divisible by any number in the range **2** to **num/2** that means it is not a prime number. This is checked, with the condition **num%k==0.**
- If this condition is true, the variable **a** assigned the value **1**, indicating that **num** is not a prime number.
- At this point, the **break** statement is used to exit the **for** loop immediately. After the execution of the **for** loop, the value of **a** is checked, if it is **0** then the message telling the user that **num** is a prime number is printed otherwise the message, **num** is not a prime number is printed.

The Output of the program is shown in Fig.

Output of Program

Program to read 10 numbers and find the sum of those that are less than 20 by using continue statement:

```
#include<stdio.h>
#include<conio.h>
void main(void)
{
    int sum=0, k, a;
    for(k=1; k<=10; k++)
    {   printf("\nEnter an integer:");
        scanf("%d", &a);
        if(a>=20)
            continue;
        sum=sum+a;
    }
    printf("\nSum of numbers less than 20 is %d", sum);
    getch();
}
```

- When the program is executed, the user enters 10 numbers one by one.
- If a number less than 20 is entered, the condition of **if** statement is false and the number is added in the sum.
- If a number greater than or equal to 20 is entered, the number is not included in the sum as the condition is true and the last statement of the loop that adds the number, is skipped using the **continue** statement.
- After ten iterations the loop terminates and the last statement of the program prints the sum of those numbers that are less than 20.

What is nested loop? Give two examples.

Nested Loops:

In C language, it is allowed to nest loops within another loop. Nested loops can be of any kind. For example, the programmer can nest a **for** loop inside a **while** loop or inside a **do while** loop. Loops can be mixed in anyway as required.

Example # 1:

Program uses nested loop to print the products of numbers as given below.

1 × 1 = 1
1 × 2 = 2
1 × 3 = 3
1 × 4 = 4
2 × 1 = 2
2 × 2 = 4
2 × 3 = 6
2 × 4 = 8
3 × 1 = 3
3 × 2 = 6
3 × 3 = 9
3 × 4 = 12

The program prints the products of number.

```
#include<stdio.h>
#include<conio.h>
void main(void)

      int j, k, prod;

      for (j=1; j<=3; j++)
          for (k=1; k<=4; k++)
        {
             prod=j*k;
             printf("\n%d x %d = %d", j, k, prod);
           }
      getch();

}
```

The Output of the program is shown in Fig.

- The loop variable **j** is assigned its initial value 1 and the nested loop

```
For (k=1; k<=4; K++)
{
    prod = j*k;
    printf("\n%2d × %2d = %2d", j, k, prod);
}
```

If executed, this calculates and displays the first four products, 1×1, 1×2, 1×3 and 1×4.
- The value of **j** is then incremented by 1 and the inner loop is executed again. This calculates and displays the next four products, 2×1, 2×2, 2×3 and 2×4.
- Finally, **j** is incremented to 3, giving the last four products, 3×1, 3×2, 3×4 and 3×4.
- In this program braces must be used in the inner **for** loop because there is more than one statement to be executed.

Example # 2:
Program uses nested loop to calculate the sum of the integers for each integer from 1 to n, where n is the value that the user of the program enters:

The execution of the program is shown in Fig.

```
the upper limit(n):8

   Integer        Sum
   1              1
   2              3
   3              6
   4              10
   5              15
   6              21
   7              28
   8              36
```

SHORT QUESTION

What is a logic gate?

- Logic gates are the basic building blocks of digital computer.
- Logic gates operate on two voltage levels and process digital signals which represent binary digits 0 and 1.

Define truth table.

- A truth table represents a digital logic circuit in table form.
- It shows how a logic circuit's output responds to all the possible combinations of the inputs using logic '1' for true and logic '0' for false.

Define Boolean function.

- A Boolean function can be transformed from an algebraic expression into a logic circuit composed for AND, OR and NOT gates.
- A Boolean function is an expression formed with binary variables, the logical operators (OR, AND and NOT), parenthesis and equal sign.
- All binary variable can take the value of 0 or 1.
- For a given value of the variables, the function can be either 0 or 1.

Example:

As an example, consider the following Boolean function.

$$F = x + y$$

The function F is equal to 0, if x = 0 and y = 0. For all the other combinations of x and y, the function will be equal to 1.

What is Karnuagh map and why is it used?

- K-map is a pictorial form of a truth table.
- It consists of square boxes called cells.
- All the possible combinations of variables involved in a Boolean function are written inside the cells in their respective positions.
- A two-variable K-map contains $2^2 = 4$ cells, three-variable $2^2 = 8$ cells and so forth.

Draw three-variable Karnaugh map for variables X, Y and Z.

- A three-variable K-map variable A, B and C is shown in Fig.

Three-variable K-map

- It consists of eight cells having two rows and four columns.
- Rows are labeled with the complement and normal form of the variable A.
- Each column is labeled with two variables B and C, in their normal or complemented form.
- Each cell contains a product term of variables A, B and C in its respected cell.
- For example, the term ABC is placed in cell that is in row A and column BC.

Rules for simplifying a three-variable K-map:

- For each term of the function, place 1 in the corresponding cell ir Karnaugh map.
- Form groups of four if possible otherwise groups of two.
- Groups can contains only 1s.
- Groups can be horizontal or vertical.
- Groups can overlap and wrap around the side of the K-map.
- If possible include each 1 in at least one group.
- Eliminate the variables that are normal and complemented form in a group and create a term for each group.
- Write the simplified function in the form of sum of terms. If a cell containing a 1 cannot be included in any group then write the full term with three variables.

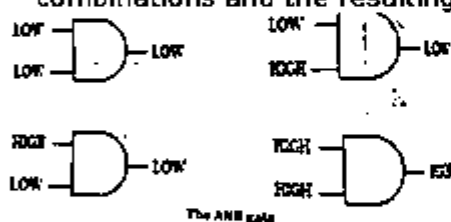
LONG QUESTIONS

Draw the graphical symbols of AND, OR, NOT, NAND and NOR gates and write their functions.

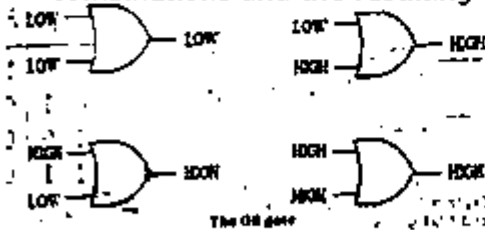
There are three basic logic gates used in digital circuits which are AND< OR and NOT gates.

AND GATE:

- The AND gate has two or more inputs that can be LOW (0) or HIGH (1).
- The output is HIGH only when al the inputs are HIGH.
- It produces a LOW output when at least one of the inputs is LOW.
- The logic operation of a two-input AND gate is shown in Fig, with all the possible input combinations and the resulting output for each.

The AND gate

OR Gate:

- The OR gate has two or more inputs, the output of an OR gate is LOW only when all the inputs are LOW.
- The output is HIGH when one or more of its inputs are HIGH
- The logic operation of a two-input OR gate is shown in Fig, with all the possible input combinations and the resulting output for each.

The OR gate

NOT Gate:

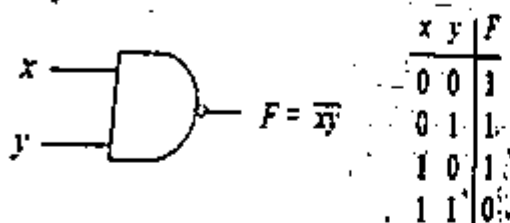
- The NOT gate performs the functions of inversion. Therefore, it is also known as inverter. It has a single input.
- The output of a NOT gate is HIGH when the input is LOW and vice versa.

LOW → HIGH   HIGH → LOW

The NOT gate

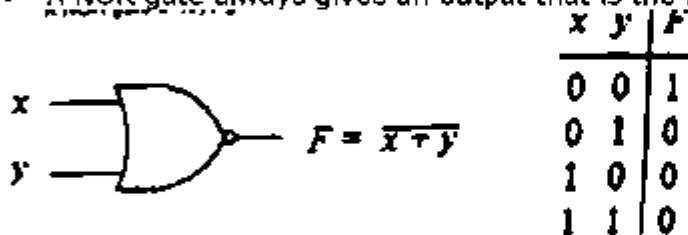
NAND Gate:

- The NAND gate combines the AND and NOT gates, such that the output will be 0 only when all the inputs are 1 as shown in Fig.
- Its logic expression is F = xy which indicates that inputs x and y are first ANDed and then the result is inverted. Inversion is indicated by a bar.
- Thus, an AND gate always produces an output that is the inverse (opposite) of and AND gate.

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

Symbol, expression and truth table of NAND gate

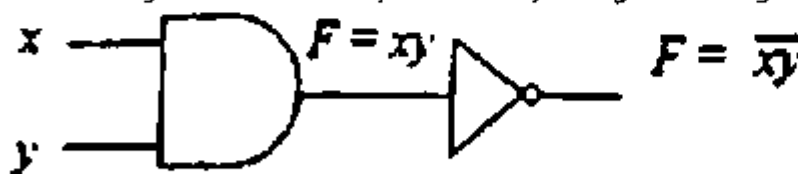
The NOR Gate:

- The NOR gate combines the OR and Not gates, such that the output will be 0 when any input is 1 as shown in FIG.
- Its logic expression is F = x + y. which indicates that x and y are first ORed and then the result inverted.
- Inversion is indicated by a bar.
- A NOR gate always gives an output that is the inverse of an OR gate.

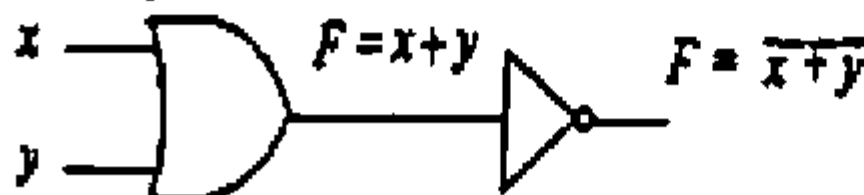
x	y	F
0	0	1
0	1	0
1	0	0
1	1	0

Symbol, expression and truth table of NOR gate

Explain how NAND and NOR gates can be created using AND, OR and NOT gates.

Creating NAND Gate Using Basic Gates:

The NAND gate can be easily created by using an AND gate and a NOT gate as shown in Fig.

Creating a NAND gate using basic gates

Creating NOR Gate Using Basic Gates:

The NOR gate can also be created in a similar way by using anOR gate and a NOT gate as shown in Fig.

Creating a NOR gate using basic gates

Draw truth table of the following Boolean functions.

- i) $F_1 = \bar{x}y\bar{z} + \bar{x}yz + xy\bar{z}$
- ii) $F_2 = \bar{x}z + y\bar{z} + xyz$
- iii) $F_3 = \bar{x}\bar{y}\bar{z} + \bar{x}\bar{y}z + \bar{x}yz + xy\bar{z}$
- iv) $F_4 = x\bar{z} + \bar{x}\bar{y}$

$F_1 = \bar{x}y\bar{z} + \bar{x}yz + xy\bar{z}$

X	Y	Z	$\bar{x}$	$\bar{y}$	$\bar{z}$	$\bar{x}y\bar{z}$	$\bar{x}yz$	$xy\bar{z}$	F_1
1	1	1	0	0	0	0	C	0	0
1	1	0	0	0	1	0	C	1	1
1	0	1	0	1	0	0	C	0	0
1	0	0	0	1	1	0	C	0	0
0	1	1	1	0	0	0	1	0	1
0	1	0	1	0	1	1	C	0	1
0	0	1	1	1	0	0	C	0	0
0	0	0	1	1	1	0	C	0	0

$F_2 = \bar{x}z + y\bar{z} + xyz$

X	Y	Z	$\bar{x}$	$\bar{y}$	$\bar{z}$	$\bar{x}z$	$y\bar{z}$	Xyz	F_2
1	1	1	0	0	0	0	C	1	1
1	1	0	0	0	1	0	1	0	1
1	0	1	0	1	0	0	C	0	0
1	0	0	0	1	1	0	C	0	0
0	1	1	1	0	0	1	C	0	1
0	1	0	1	0	1	0	1	0	1
0	0	1	1	1	0	1	C	0	1
0	0	0	1	1	1	0	C	0	0

$F_3 = \bar{x}\bar{y}\bar{z} + \bar{x}\bar{y}z + \bar{x}yz + xy\bar{z}$

X	Y	Z	$\bar{x}$	$\bar{y}$	$\bar{z}$	$\bar{x}\bar{y}\bar{z}$	$\bar{x}\bar{y}z$	$\bar{x}yz$	$xy\bar{z}$	F_3
1	1	1	0	0	0	0	0	0	0	0
1	1	0	0	0	1	0	0	0	1	1
1	0	1	0	1	0	0	0	0	0	0
1	0	0	0	1	1	0	0	0	0	0
0	1	1	1	0	0	0	0	1	0	1
0	1	0	1	0	1	0	0	0	0	0
0	0	1	1	1	0	0	1	0	0	1
0	0	0	1	1	1	1	0	0	0	1

$F_4 = x\bar{z} + \bar{x}\bar{y}$

X	Y	Z	$\bar{x}$	$\bar{y}$	$\bar{z}$	$x\bar{z}$	$\bar{x}\bar{y}$	F_4
1	1	1	0	0	0	0	0	0
1	1	0	0	0	1	1	0	1
1	0	1	0	1	0	0	0	0
1	0	0	0	1	1	1	0	1
0	1	1	1	0	0	0	0	0
0	1	0	1	0	1	0	0	0
0	0	1	1	1	0	0	1	1
0	0	0	1	1	1	0	1	1

Simplify the Boolean functions of Question 5 using Karnaugh map.

Draw the logic circuits of the following Boolean functions.

===

SHORT QUESTION

Differentiate between website and Web server.

Website:

- A website is a collection of related Web pages hosted on a Web server.
- It is accessible through an internet address known as Uniform Resource Locator (URL).
- A URL is typed in Web browser and ENTER button is pressed to view it.

Web Server:

- A Web server is a computer that makes Web pages available through the internet.
- Web servers are used to host websites, data storage and for running applications.
- Web servers deliver HTML documents when a user types a URL in browser.
- Any computer can be turned into a Web server by installing server software and connecting it to the Internet.

Describe how a search engine is used for searching information on the Internet.

Search Engine:

A search engine is a website or software that allows people to find information on the World Wide Web (WWW).

Use of Search Engine:

- Finding exactly what the user is looking for is not easy from billions of Web pages.
- Therefore, search engine have been designed to help users find specific information on the Web.
- Users type in one or more keywords in a search engine.
- The search engine will look for matches found from all over the Web.
- User will see the total number of matches found and then the first ten sites that most closely match the keywords.
- The information shown for each website includes a title and a brief description.
- Most popular search engines are Yahoo! And Google.

Define URL and Web hosting.

Uniform Resource Locator (URL):

Uniform Resource Locator (URL) is an Internet address that identifies a website.

Web Hosting:

Web Hosting is a service that uploads a website on a Web server and makes it available for computer users.

Describe HTML.

- HTML is a mark-up language use to create Web pages.
- A browser such as internet Explorer or Google Chrome is used to read HTML document and display it on the screen as Web pages.

===

Hypertext Mark-Up Language:

- HTML stands for Hypertext Mark-up Language.
- It is called mark-up language because it uses mark-up tags that tell the Web browser how to display the Web page.
- It is the language of Internet's World Wide Web.
- Websites and Web pages are written in HTML.
- It is used to create hypertext documents that bring together text, pictures, sounds, video clips and links all in one place.
- HTML files are plain text files, so they can be created using a simple editor such as Notepad or WordPad.

LONG QUESTION

Describe any four types of websites.

1) Web Portals:

- Web portal is a website that offers a large variety of services.
- These include online shopping malls, news, stock prices, e-mail, search engines, etc.
- For example, a school Web portal delivers information about the school's historical background, admission requirements, school curriculum, tuition fees structure, school news, student results, etc.

2) News Websites:

- News websites provide information about current events and opinions.
- These sites publish news stories and let their visitor's voice be heard.
- These sites provide their visitors a way to get their thoughts and views published.
- Very often at the end of news stories, visitors are asked to share their experience or knowledge about the topic.
- Therefore, a link for the readers comment is provided for feedback.
- For example www.paktribune.com, www.dailytimes.com etc.

3) Information Websites:

- Information websites provide information on any topic in the form of text, graphics, sounds and videos.
- For example, www.wikipedia.org is an informational website that provides information on thousands of articles.
- It allows users to contribute and edit articles as well.

4) Educational Websites:

- Educational websites are created for educational purpose.
- These sites contain animation, slide presentation and tutorials to educate people on various topics, information is presented in a very well organized way.

- The purpose of any educational website is to impart knowledge to people who are looking for better understanding of a topic and pursue knowledge.
- For example www.downloadclassnotes.com etc.

Write the HTML tags for the following?

i)	**Paragraph**	**vii)**	**Center text**
ii)	**Heading**	**viii)**	**Strike out**
iii)	**Bold**	**ix)**	**Superscript**
iv)	**Underline**	**x)**	**Subscript**
v)	**Italic**	**xi)**	**Font size, color and typeface**
vi)	**Inserting Line Breaks**	**xii)**	**Inserting Spaces**

i) Paragraph:
The <p></p> tags are used to define paragraph. HTML automatically adds an extra blank line before and after a paragraph.

ii) Heading:
There is six heading tags, <h1></h1> to <h6></h6>. The <h1></h1> tags are used to specify the largest heading and <h6></h6> tags specify the smallest.

iii) Bold:
**Bold **
These tags will make the text bold that is within the tags.
For example: Chapter 5 Output Devices

iv) Underline:
Underline <u></u>
These tags will underline the text that is within the tags.
For example: <u>Please Note:</u>

v) Italic:
Italic <i></i>
These tags are used to make the text italic that is within that tags.
For Example: <i>I Love Pakistan</i>

vi) Center text:
Center <center></center>
These tags will make the text centered that is within the tags, the text will appear in the middle of the left and right margins.
For example: <center>UNIT 2 PROGRAMMING IN C</center>

Vii) Strike out:
Strike out<strike></strike>
These tags will put a line right through the center of the text that is within the tags. This is known as strike-out.

For Example: <strike>Obsolete Computer Devices</strike>

viii) Superscript:

**Superscript **

These tags are used to convert the text to superscript that is within the tags.
For example: X_Y

ix) Subscript:

**Subscript **

These tags are used to convert the text to subscript that is within the tags.
For example: X₅

x) Font size, color and typeface:

Font Size

These tags are used to change the font size. Replace the ? symbol with a number in the range 1 to 7. One is the smallest and seven is the largest font size.
For example: <Types of Computers

**Font Color **

These tags are used to change the color of text that is within the tags. Replace the ? symbol with color such as black, blue, brown, gray, green, maroon, orange, pink, red, white, yellow, etc.
For example: <Laptop Computer
Result: Laptop Computer

**Font Face **

These tags are used to change the font face of text that is within the tags. Replace the ? Symbol with font face such as arial, courier, Calibri, new times roman, etc.
For example: FORMATTING TAGS

xi) Inserting Line Breaks:

The <br> tag is line break tag. It is used to end a line wherever it is placed and it does not have a closing tag.

xii) Inserting Spaces:

Very often, spaces are needed in HTML documents. If the user inserts more than one space anywhere in a document, the browser will show only one space, rest of the spaces will be truncated. If the user wants to have many spaces in a HTML document then the   character entity must be used. Some characters have special meaning in HTML, like   (non-breaking space) and <, these are known as character entities.

Describe how background color and image are applied to Web page.

To apply a background color in a page, insert the background color attribute and for foreground color, insert text attribute in the <body> tag as shown in Fig.

```
<html>
<head>
<title>Background & Foreground Color</title>
</head>
<body bgcolor=red text=yellow>
<font size=4>
Computers are used in all types of businesses to<br>
improve productivity. They help in successfully<br>
running business activities. They are used to<br>
prepare business documents, reports, invoices,<br>
charts, etc. Computers help in staying in contact<br>
with employees and customers.</font>
</body>
</html>
```

Applying Background Image:

An image can be set as background of a page using the background attribute in the <body> tag.

<body background="image.jpg">

If the image is smaller than the browser window then it will repeat itself till the entire window is filled.

The HTML document in Fig, demonstrates the use of image file computer.jpg as background of a Web page.

```
        Background & Foreground Color

        background=computer.jpg text=yellow
        size=4
Computers are used in all types of businesses to
improve productivity. They help in successfully
running business activities. They are used to
prepare business documents, reports, invoices,
charts, etc. Computers help in staying in contact
with employees and customers.
```

Create an HTML document that contains a graphical hyperlink.

Creating a Graphical Hyperlink:

The syntax for creating a graphical hyperlink to another Web page is

<image scr="name of image file>

For example:

<img src="image.jpg">

It will display the image.jpg image file in the browser window. When the reader moves the mouse pointer over the image, it will change to a small hand and clicking on the image will open the Federal Board website that has the URL www.fbise.edu.pk

The HTML document in Fig, demonstrates how to create a graphical hyperlink.

```
<html>
<head>
<title>Inserting Image Anchor</title>
</head>
<body>
<a href="http://www.fbise.edu.pk"><img src="image.jpg"></a>
<br><br>
<font size=3>Click on the image to visit Federal Board Web site</font>
</body>
</html>
```

Describe the tags used for creating a table in HTML.

Tables are very often used in Web pages to present information I an organized manner which is easy to read and understand by the user.

A table consists of rows and columns. The following tags are used for creating a table.

<table></table>: These tags are used to create a table.

<tr></tr>: These tags define a row.

<td></td>: These tags define data cell. Column of each row is called data cell. A data cell can contain text, image, paragraph, etc.

The HTML document for creating the following table (Fig.) for date sheet of examination is shown in Fig. This table has 5 rows and 4 columns.

In the HTML document of Fig, The tag

<table border=1 cellspacing=3 cellpadding=8>

It tells the browser to display the table with border and to have cell spacing equal to 3 pixels and cellpadding to 8 pixels. Cell spacing is the space between the cells of the table and cell padding is the space between the text and the border of the cell.

The space between the cells and also the space between the text and the border of the cells has decreased to default values.

In the HTML document, the code for creating the heading of each column of second row is given as below.

```
<tr>
<td>S.No</td>
<td>Date</td>
<td>Class IX</td>
<td>Class X</td>
</tr>
```

The text of all the four cells of second row is within the <tr></tr> tags and the text of each cell is within the <td></td> tags. The text of the next three rows is also defined in the same way.

The code for the first row that contains the title is:

```
<tr>
<td colspan=4 align=center>
<b>FINAL EXAMINATIONS 2011-2012</b></td>
</tr>
```

Here, the colspan attribute is used to combine the four columns and the align attribute centers the table. The user can also use the rowspan attribute to combine the cells of rows of a table. The align attribute is used to align the title in the center of the row. Title can also be aligned to the left or right.